Topic Popularity Prediction with Sentiment Time Series on Short Text based Social Media*

Jinning Li, Qiang Zhang, Jiayi Xu, Xiaofeng Gao†, Guihai Chen
Shanghai Key Laboratory of Scalable Computing and Systems,
Department of Computer Science and Engineering,
Shanghai Jiao Tong University, Shanghai, 200240, China
{lijinning, zhangqiang2016, xujiayi925}@sjtu.edu.cn, {gao-xf, gchen}@cs.sjtu.edu.cn

Abstract

The analysis and prediction of topic popularity is an crucial task on short text based social media. It predicts the trend of a given topic according to logged historical data. However, few of previous models apply public sentiment analysis to facilitate popularity prediction. In this paper, we propose a novel framework to better predict topic popularity utilizing sentiment factors. This framework includes a semantic-aware popularity quantification metric to capture topic popularity, A novel tree-like deep learning architecture combining LSTM and CNN for sentiment analysis, and a sentiment-aware time series prediction model. Comparative experiments on Twitter datasets prove that our model outperforms other models by reducing the error of popularity prediction. Experiments also prove the effectiveness of auxiliary sentiment features by improving the accuracy of non-sentiment models.

1 Introduction

Social media have become key parts of the modern lifestyle. They are not limited to well-known platforms such as Facebook and Twitter. Many news and video sharing platforms such as BBC News and YouTube also provide social interaction services for their users. These social media are continuously generating a huge amount of text-based information, revealing both individual preferences and public attention. On these social media, users usually tend to discuss trending topics. For example, release of the movie *Avengers: Infinity War* has become a popular topic on both Twitter and YouTube.

*This work is supported by the National Key R&D Program of China (2018YFB1004703), the National Natural Science Foundation of China (61872238, 61672353), the Shanghai Science and Technology Fund (17510740200), the CCF-Tencent Open Research Fund (RAGR20170114), and the Huawei Innovation Research Program (HO2018085286).

†Corresponding author.

Predicting the popularity of trending topics is an important part of user behavioral analysis on social media. Accurate popularity prediction can help companies improve service effectiveness and user experience by providing better content recommendation, online advertising and information search services. Understanding and predicting the popularity of topics are also necessary for specific stakeholders such as consulting firms to investigate real-time public sentiments and opinions. However, there are several major **challenges** in building an accurate popularity prediction model.

Table 1: Sample tweets about the movie *Avengers: Infinity War*. Words, emojis, URLs are included.

1	I love Avengers Infinity War moive 😍
2	AVENGERS INFINITY WAR FREE: [URL]
3	RT @OnePlus_IN: The OnePlus 6 Goes Silk White and the Avengers Limited Edition Goes Live.

The first challenge comes from the quantification of popularity. Typically, previous works usually use simple considerations such as the number of retweets, likes and views [1, 2]. However, on the one hand, these methods can not semantically solve the **problems of noise and ambiguity**. For example, Table 1 shows three sample tweets about the topic *Avengers: Infinity War*. The quantification methods based on retweets will treat these three tweets exactly the same. However, the third tweet is actually discussing a limited edition of mobile phones instead of the movie. On the other hand, many social media do not collect or publish the data of forwarding, likes, and views. Therefore a more accurate and robust **semantic-aware quantification model** is needed.

The second challenge is whether other features can be taken into consideration to improve the performance of popularity prediction. Through studying the twitter dataset, we find an interesting relation between topic popularity and the public sentiment towards it. The topic with high popularity tends to gain strong public

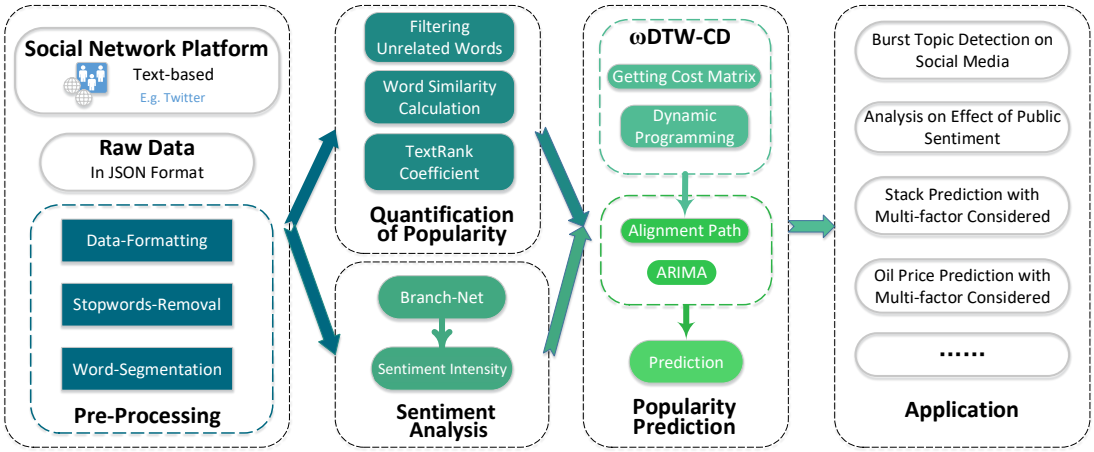

Figure 1: Framework of SENTI2POP. The input is text-based data from social media. After the pre-processing procedure, quantification of topic popularity and sentiment analysis are respectively implemented and fed to our topic prediction model. It is worth mentioning that we can not only use this framework for social media topic popularity prediction, but also apply it to many other application domains such as burst topic detection or stack prediction.

sentiment, whether the public opinions are positive, negative or divided in two confronted emotion groups. On the other hand, an emotion-intensive topic is prone to obtain more discussion and attention, thus becoming more popular. **Public sentiment towards an topic is closely associated with its popularity.**

The third challenge is the predicting model. After properly quantifying the topic popularity and capturing the sentimental aspect, two time series of features, *Popularity Time Series* (PTS) and *Sentiment Time Series* (STS) will be obtained. Temporally, these two series are strongly related. How to **develop a predicting model, which can well capture the relation between PTS and STS** to achieve higher predictiong accuracy, remains a challenge.

Based on these considerations, we propose a novel framework, SENTI2POP, to better predict topic popularity, which is visualized in Fig. 1.

To quantify the topic popularity, we propose a **semantic-aware quantification model**, namely, *Term Frequency with Text Rank Coefficient* (TF-TRC) in Section 3. This model is capable of capturing the semantic relation between words and the topic, based on the frequency of words which are relevant to a selected topic. Word2Vec [3] and TextRank [4] algorithms are adopted to calculate the importance of words. To improve the efficiency of TF-TRC, we further propose a cut-off mechanism based on Zipf's law [5] to filter out unrelated words.

The hybrid architecture of Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) have been proved to be effective in sentence encoding [6]. Inspired by this hybriding idea, we propose a **cascaded, tree-like neural network architecture**, namely, *Branch-Net* to **better capture sentiment features**, which will be introduced in Section 4.1. Emojis and subjective words in the tweets are used to build the training set for Branch-Net.

Dynamic time warping algorithm [7, 8] is widely used in automatic speech recognition to match two voice series. To predict the popularity, we first propose a temporal matching algorithm adapted from the dynamic time warping to capture the relation between PTS and STS, namely, *Weighted Dynamic Time Warping with Compound Distance* (wDTW-CD). We then propose a *Dual Autoregressive Integrated Moving Average (Dual ARIMA)* algorithm to **predict the lead-lag relation in the future**. The predicted relation is used to estimate the future topic popularity combining the features of PTS and STS.

Comparative experiments are carried out to evaluate the performance of Senti2Pop. Widely used prediction models, both with and without sentiment information, are compared with SENTI2POP model. Results show that **Senti2Pop outperforms other models with the minimum prediction error**. Results also show that by using the sentiment features extracted by Branch-Net, the **prediction errors of non-sentiment models improve** 21% on average. Experiments on SST and CMRD datasets also shows **the effectiveness of Branch-Net on sentiment analysis**.

2 Overview and Problem Formulation

SENTI2POP is a predicting framework which aims at quantification and prediction of future topic popularity, based on historical records of text-based data on social media. It receives historical text data of a topic from a given specific social medium. Every piece of text data, named as a record, contains both text and its corresponding time stamp. For example, each tweet in Table. 1 is a record.

In the perspective of time, the historical time period, denoted by T, is the collection of time stamps covering the historical data. To discretize the problem, we set a time interval Δt. The historical time period T is separated to many time periods t_k, $T =< t_1, t_2, \cdots, t_n >$, where $t_k = t_{k-1} + \Delta t, k \in \mathbb{N}$.

After the preprocessing, unrelated contents such as URLs, stopwords, and punctuation are filtered out. Word segmentation is applied to separate words from the sentences. A special kind of unicode word, emoji, is preserved for sentiment analysis.

We use r_t^i to represent the ith record at time t. Every record is made up of a series of words, $r_t^i = w_t^1, w_t^2, \cdots$, where w_t^j denotes the jth word in the record at time period t. For example, after preprocessing, the tweets in Table 1 are transformed into Table 2.

Table 2: Sample tweets after preprocessing

#	Words	Emojis
1	[love, avengers, infinity, war, moive]	😍
2	[avengers, infinity, war, free]	-
3	[oneplus, go, silk, white, avengers, limit, edition, live, check, out]	-

After that, all the records at time t constitute the recording set $R_t = \{r_t^1, r_t^2, \cdots\}$. This tree-like data structure is shown in Fig. 2.

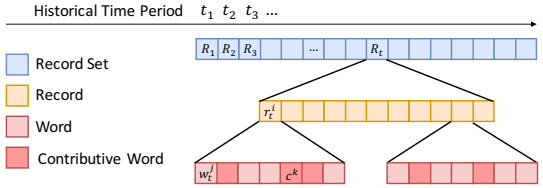

Figure 2: Data structure of the concepts. Arrows above represent the historical time period when the data is recorded. t_k is the discretized time period. The squares in colors represent different concepts.

The first part of SENTI2POP is the quantification of popularity P_t at every time period t and the definition of *Popularity Time Series* (PTS), which will be introduced in Section 3. In SENTI2POP, we introduce a novel method, *Term Frequency with Text Rank Coefficient* (TF-TRC), to quantify the popularity P_t. Then, Zipf's Law [5] are applied to filter out unrelated words. The words left are named as contributive words c_t^i. All the contributive words at time period t is represented by C_t. After that, we build a undirected graph where vertices represent contributive words c_t^i and edges represent the semantic similarity ξ between c_t^i. By applying the PageRank [9] algorithm on the undirected graph, the *Text Rank coefficient* $TR(\cdot)$ are induced in Eqn. 3.2. Then, the popularity P_t is defined by Eqn. 3.1, which is related to the frequency $fre(\cdot)$ of c_t^i and the $TR(\cdot)$ value. PTS is defined as the time series of P_t.

To quantify the *Sentiment Intensity* S_t at time period t and define the *Sentiment Time Series* (STS) of the topic, we introduces a novel neural network, *Branch-Net*. Branch-Net is based on a hybrid architecture [6] combining Bi-directional Long-Short Term Memory (Bi-LSTM) [10] and convolutional neural network [11] to learn the mapping from text to its sentiment intensity.

We first define the Sentiment Intensity $\varepsilon(r_t^i)$ of a record r_t^i by the appearance of emojis and the subjectivity of words labeled in the MPQA subjectivity lexicon [12]. With the labeled $\varepsilon(r_t^i)$, we use Branch-Net to learn the mapping from the records to their sentiment intensity. The sentiment intensity of the topic S_t is defined in Section 4. STS is the time series of S_t at each time period t.

The objective of SENTI2POP is to predict the topic popularity in the future leveraging the given historical records. There is a strong relationship between public sentiment intensity S and the topic popularity P. We propose wDTW-CD and *Dual ARIMA* algorithm to better leverage this relationship and predict the topic popularity. The input of Dual ARIMA is the PTS $P =< P_1, P_2, \cdots, P_n >$ and STS $S =< S_1, S_2, \cdots, S_n >$. By applying wDTW-CD, A temporal lead-lag series l and a vertical distance series v are obtained. Then, Dual ARIMA is applied to predict and extend these two series temporally. At last, the predicted popularity is calculated according to Eqn. 5.6.

3 Term Frequency with Text Rank Coefficient (TF-TRC)

There are various kinds of quantification metrics for topic popularity, most of which only analyze popularity in the level of tweets. This means the features such as views, likes and retweets are taken into account. However, semantic noise and ambiguity problems will limited the performance of tweet-level metrics. In this section, we are going to introduce a semantic-aware popularity quantification model in word level, which is more generalized and accurate. TF-TRC metric quantifies the popularity of a topic by combining the frequency and the semantic relationship between the

words and the topic.

In TF-TRC metric, the popularity of the topic is defined as:

$$P_t = \sum_{i=0}^{|C_t|-1} \left[fre(c_t^i) \cdot TR(c^i) \right], \tag{3.1}$$

where P_t is the popularity at time period t. c_t^i is the contributive word in the set of contributive word C introduced in Section 3.1. $fre(c_t^i)$ is the frequency of c_t^i. $TR(c^i)$ is the Text Rank coefficient of c^i, which will be introduced in Section 3.2. The Text Rank coefficient is a time-independent variable.

Intuitively, $fre(c_t^i)$ is the frequency of a contributive word used at time t. $TR(c^i)$ represents the relation between the word c^i and the topic selected. A higher frequency means larger scale of contributive words is used to discuss the topic, such that the popularity of this topic becomes higher. However, the importance of these words are different in terms of their relation to the topic. So $TR(\cdot)$ coefficient is applied to balance the importance of words.

When a tweet is retweeted, the initial tweet will also be repeated with an additional RT tag, just like the third sample in Table 1. This means the frequency of words $fre(\cdot)$ in the initial tweet will increase, which is similar to the tweet-level retweet counting methods. Actually, in the extreme case where every tweet contains only 1 word, value of $TR(\cdot)$ is always equal to 1, such that TF-TRC will degenerate into the retweet-counting metric.

3.1 Distribution of Word frequency There are tons of unrelated noisy words in all the tweets relative to a certain topic, which will result in higher computational complexity and lower accuracy. We propose a cut-off threshold mechanism to eliminate those noisy words and only consider the remaining words. Those remaining words are define as *contributive words* C.

For example, let's consider all the tweets relative to the topic *Trump* and *gun control* respectively, and count their words' occurrence frequency. Results are shown in Fig. 3. We find the word frequency obey the long tail distribution and Zipf's Law [5].

According to this characteristic, we set a threshold to filter out unrelated words based on Zipf's Law, which reduce the whole complexity of the framework.

3.2 TextRank Coefficient In this section, we introduce the second term in Eqn. 3.1, TextRank coefficient, to describe the importance of a word in terms of the selected topic.

The original PageRank algorithm is calculated on a directed graph where each vertex is a web page and each

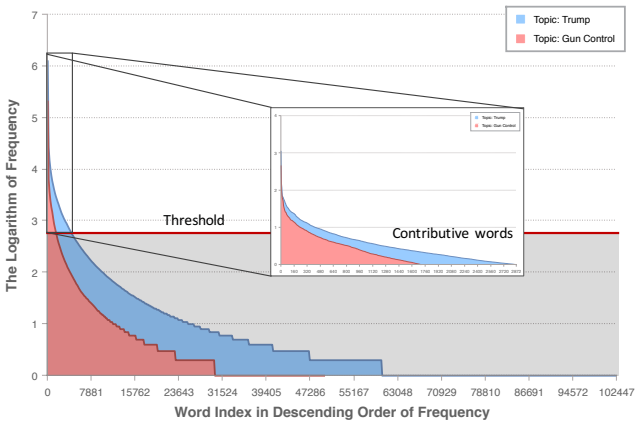

Figure 3: Filtering according to Zipf's law. The words whose frequency is larger than the threshold is collected as contributive words.

directed edge represents a hyper link. In our task, each vertex represents a word. The relations between vertices are bidirectional, which represents word similarity. In this way, we build a weighted undirected graph to run PageRank algorithm.

First of all, we need to define the weights of edges in the undirected graph. These weights are supposed to reflect the relationships between two connected words. We introduce the concept *similarity* between words w_t^i and w_t^j, denoted as $\xi(w_t^i, w_t^j)$ to represent these relations.

A graph is constructed for a topic's records R_t at each time period, in which each vertex represents a distinct word and the edge refers to their similarity $\xi(w_t^i, w_t^j)$. We run PageRank algorithm on the graph to get the importance of each contributive word $TR(w_i)$. The formula for TextRank is defined as

$$TR(w_t^i) = \frac{1-\theta}{|C|} + \theta \cdot \sum_{j \to i} \frac{\xi(w_t^i w_t^j)}{\sum\limits_{k \to j} \xi(w_t^k, w_t^j)} \cdot TR(w_t^j), \tag{3.2}$$

where the factor θ, ranging from 0 to 1, is the probability to continue to random surf follow the edges, since the graph can be an imperfect graph and face potential dead-ends and spider-straps problems in practice. θ is usually set to be 0.85 empirically [4]. The number of all contributive words are denoted as $|C|$, and $j \to i$ refers to words that is adjacent to word w_i.

The similarity between words is determined by their semantical and lexical relationships, so it can be denoted as $\xi(w_t^i, w_t^j) = \gamma \cdot \vartheta(w_t^i, w_t^j) + (1-\gamma) \cdot \eta(w_t^i, w_t^j)$, where γ is the parameter to make our model flexible and general to different languages. $\vartheta(w_t^i, w_t^j), \eta(w_t^i, w_t^j)$ represent the semantic and lexical relations between word w_t^i and w_t^j respectively. $\eta(w_t^i, w_t^j)$ is equal to the number of common letters in w_t^i and w_t^j, divided by the total

number of letters of them. The semantic relations are defined by:

(3.3)
$$\vartheta(w_i, w_j) = \beta \cos(\boldsymbol{W}_i^{\mathbb{R}}, \boldsymbol{W}_j^{\mathbb{R}}) + (1-\beta)\cos(\boldsymbol{W}_i^{\mathbb{D}}, \boldsymbol{W}_j^{\mathbb{D}}),$$

where $\boldsymbol{W_i}, \boldsymbol{W_i}$ are vectors mapped from word w_t^i, w_t^j according to Word2Vec. $\mathbb{R}, \mathbb{D}$ are two corpora integrated to comprehensively reflect the event characteristics. $\mathbb{R}$ is an topic corpus from our dataset while $\mathbb{D}$ is a supplementary corpus extracted from Wikipedia Dump. Parameter β is related to the two corpora and determines which term to emphasize.

After the Text Rank coefficient being calculated with Word2Vec and TextRank algorithm, the popularity of a topic is induced according to Eqn. 3.1.

4 Branch-Net for Sentiment Analysis

Popularity Time Series (PTS) can be generated by the TF-TRC metric introduced In Section 3. With this time series, many previous prediction models are already capable of predicting future value [13].

However, the performance of this kind of prediction is quite limited because there will be many social factors leading to a sharp change of topic popularity. For example, the emergencies and reports of media related to the given topic. However, before these topics break out, public sentiment intensity will have a rapid change. This is an important clue which can be leveraged to predict the popularity more precisely.

To solve this problem, SENTI2POP model combines the information of public sentiment using a hybrid neural network to optimize the prediction of topic popularity. On social network, a record usually contain the information of time stamp, text, and also a special kind of unicode words, emojis. In the text data, the subjectivity also varies from word to word. These are the most important two information related to their sentiment intensity. The appearance of different emojis and the subjectivity words can be used to evaluate the sentiment intensity of a document.

The emojis in the texts are the most characteristic and direct feature to represent the sentiment intensity, however, only a few of all the records will contain emojis. As a result, it is really hard to evaluate the sentiment intensity of these records. And the original subjectivity lexicon method which ignores the semantic information, may also fail to predict the sentiment intensity. To solve these two problems, we propose a novel architecture, Branch-Net in Fig. 4 to learn the mappings. Branch-Net is based on a hybrid architecture [6] of Bi-LSTM [10] and CNN [11]. After training the Branch-Net with billions of records, this end-to-end network is capable of figuring out the potential semantic information and mapping it to the corresponding sentiment intensity.

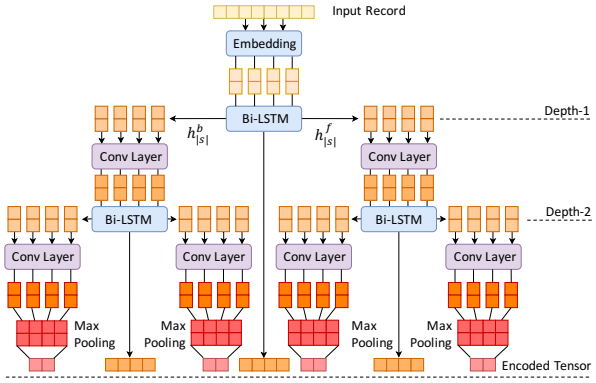

Figure 4: The architecture of the Branch-Net. The Branch-Net includes several Bi-LSTM layers, convolutional layers, pooling layers, skip connections and an embedding layer. Fully connected layers and softmax function are applied after the encoded tensor for different classification tasks.

Shown in Fig. 4, the input records are sets of words, which is then mapped to a vector space with an embedding layer. There will be a branch at every Bi-LSTM layer, representing the backward and forward information. After each branch, a convolutional layer is applied to further encode the information. Skip connections [14] are also applied on every Bi-LSTM layer.

To make the pre-trained Branch-Net adaptive to the taget twitter corpus, we have a metric to evaluate the sentiment intensity ε of each record r_t^i. A probable solution is to manually tag the sentiment intensity of every piece of record. However, the manual tagging will cost much time and manpower resource. In this paper, we use the sentiments expressed by emojis and subjective words to quantify the ε of their original records.

We use the emojis appearing in the text to define the sentiment intensity. ω_e is the sentiment vector of the emojis. The kth dimension of ω_e, $0 \le \omega_e^k \le 1$, is the sentiment intensity of the kth emoji. $\phi_{r_t^i}$ is the indicator vector of the record r_t^i. $\phi_{r_t^i}^k = 1$ if the kth emoji exists in r_t^i, and $\phi_{r_t^i}^k = 0$ otherwise. Then, the sentiment intensity $S(r_t^i)$ is defined as:

(4.4)
$$\varepsilon(r_t^i) = \frac{\omega_e \cdot \phi_{r_t^i}}{|\phi_{r_t^i}|}$$

4.1 Branch-Net Convolutional neural networks (CNN) are widely used to many NLP tasks. The strength of CNN is to process and find the semantic information of local n-grams. However, the CNN achitecture is not capable to depict the overall feature

of a series of words. On the contrary, Recurrent architectures such as Bi-LSTM are proved to be excellent when processing the sentences with long-range dependency, which means the Bi-LSTM can learn to forget the useless information of historical words and remember the important ones. Besides, to train a deep neural network, skip connection is a useful trick to improve the performance of neural networks and help the neural network to combine both of the original and deep-encoded information.

In our task, the text data is short, so the analysis of local n-grams becomes quite important. At the same time, the sentences in a record is strongly related in logic. The long term dependency of the sentences, both in the backward and forward direction, should be well addressed. So, based on the ideas of Bi-LSTM and CNN, we proposed a novel hybriding architecture, named Branch-Net, to process the word series of every record. Its architecture is shown in Fig. 4.

5 Dual ARIMA for Popularity Prediction

In this part, we are going to introduce Dual ARIMA algorithm to predict popularity according to historical PTS and STS time series.

ARIMA is the most classical method used to predict time series. The performance of the original ARIMA is limited when the distribution of the time series is non-stationary.

To improve the performance, we first propose the wDTW-CD algorithm to implement an temporally matching between PTS and STS. After that, wDTW-CD will produce an extended temporal lead-lag series l' and vertical distance series v'. Then, instead of only using the information of single time series, Dual ARIMA will esitimate the popularity in the future according to these two series, which obey the stationarity distribution. At the same, Dual ARIMA also automatically combines the result of single ARIMA to achieve greater robustness.

5.1 wDTW-CD Dynamic time warpping (DTW) algorithm is widely used in speech recognition tasks. However, with only the local minimum cost considered but ignoring the global optimum, classic DTW with Euclidean distance may suffer from far-match and singularity problems when aligning PTS and STS. The Far-Match means the the alignment of PTS and STS's data points are temporally too far away. The Singularity problem means that a single point is aligned to too many points of another time series. So, a more robust and reasonable adaptation should be discussed.

In this paper, we introduce the compound distance and temporal weight to solve the Far-Match and Singularity problems of classic DTW.

Compound Distance: The compound distance can help solve the Singularity problem. The definition of compound distance is given by $d^C_{i,j} = \sqrt{d^E_{i,j} \cdot d^D_{i,j}}$, where $d^E_{i,j}$ is the Euclidean distance, $d^D_{i,j}$ is the derivative distance. Trend features are described by the estimated derivative of data points on PTS and STS, denoted by d^D. According to [8], d^D generated by $d^D_{i,j} = |D(P_i) - D(S_j)|$, where $D(P_i)$ and $D(S_j)$ are the estimated derivative in the point P_i and S_j of PTS and STS.

Temporal Weight: The temporal weight is introduced to the wDTW-CD to solve the Far-Match problem. We use a sigmoid-like function to define temporal weight $\omega_{i,j} = \frac{1}{1+e^{-\eta(|i-j|-m)}}$, where η decides the overall penalty level, which can be adjusted for different patterns of PTS and STS. Factor m is a prior estimated time difference.Intuitively, if the temporal distance between P_i and S_j is larger than m, the alignment will obtain a relatively high penalty. And a larger η will result in a higher penalty.

Within the compound distance and temporal weight defined, the cost of dynamic time warpping process is given by $d_{i,j} = d^C_{i,j} + \omega_{i,j}$. Then, we use P and S to represent the PTS and STS. Suppose $P = \{P_1, P_2, \cdots, P_n\}$ and $S = \{S_1, S_2, \cdots, S_n\}$, which are temporally warped but not aligned yet. The goal of DTW is to find an optimal warping path to align these two sequences. $d_{i,j}$ is the distance between P_i and S_j.

After applying DTW algorithm, the alignment path Z can be generated, which is visualized graphically in a heatmap shown in Fig. 6. The aligned PTS and STS are visualized in Fig. 5 providing a intuitive perspectives to the pattern of a certain topic's popularity and sentiment intensity.

5.2 Predict with Dual ARIMA The ARIMA model can be used to predict the time series and has been proved to have a good performance. However, the time series processed by the ARIMA model is required to have a stationarity or at least have a stationarity after making an n-order difference.

With Deniel test, we find that PTS and STS are always not or only weakly stationary. So it is not suitable to use the traditional ARIMA to predict the time series directly. Instead, Dual ARIMA model uses an indirect method. It use the temporal lead-lag series l and vertical distance series v induced by the wDTW-CD as the target series, which are proved to be stationary with the Deniel test. Then the popularity is estimated using the extended l' and v'.

Temporal lead-lag series l is induced by the DTW path Z introduced in Section 5.1 $l_t = z^{(1)}_t - z^{(0)}_t$, where

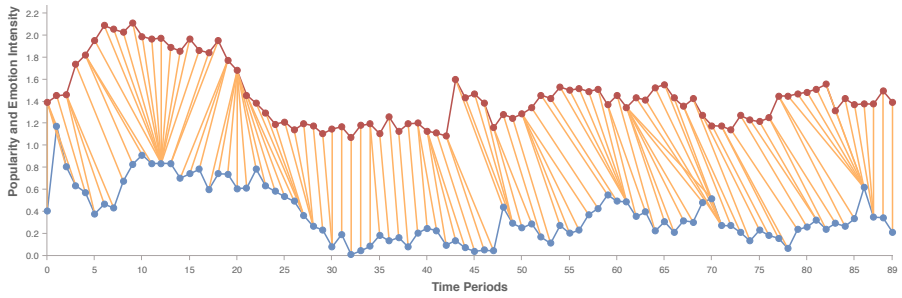

Figure 5: Part of the alignment result with wDTW-CD of the topic *Gun Control*. The red line is the STS which the blue line is the PTS. We can find that the overall trend of the aligning lines inclines to the STS, which means the emotion change leads the trend of popularity in this case.

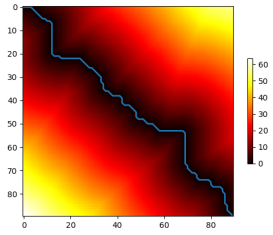

Figure 6: The heatmap of wDTW-CD. The blue line represents the alignment path Z in Section 5.1

$z_t^{(i)}$ means the i-th dimension in the path z_t.

Similarly, the vertical distance series v is defined as $v_t = P(z_t^{(1)}) - P(z_t^{(0)})$, where the $P(z_t^{(i)})$ is the popularity at time period $z_t^{(i)}$. Then, we apply ARIMA to predict and extend l and v.

The ARIMA forecasting equation for a stationary time series is a regression-type equation in which the predictors consist of lags of the dependent variable and lags of the forecast errors. For the non-seasonal ARIMA(p, d, q) model, the projected value is computed by $(x_t - \mu) - \phi_1(x_{t-1} - \mu) - \cdots - \phi(x_{t-p} - \mu) = \epsilon_t - \theta_1\epsilon_{t-1} - \cdots - \theta_q\epsilon_{t-q}$, where p represents the number of AR (Auto-Regressive) terms and q represents the number of MA (Moving Average) terms. We use the AIC law to measure the value of p and q:

$$\min \ AIC = n \ln \hat{\sigma}_\epsilon^2 + 2(p + q + 2). \tag{5.5}$$

We use ARIMA to predict the series of P, l and v. After P', l', and v' are the predicted value by single ARIMA, the result of Dual ARIMA is given by:

$$P(t) = \alpha\,[S(t + l'(t)) - v'(t)] + (1 - \alpha)P'(t), \tag{5.6}$$

where $P(t)$ is the topic popularity. $S(t)$ is the sentiment intensity. $\alpha = 1 - \frac{2}{\pi}\beta$ and $\beta = \max(-\arctan\frac{v'(t)}{l'(t)}, 0)$. $P(t)$ can then be calculated by this equation. Procedure of this prediction is shown in Fig. 7.

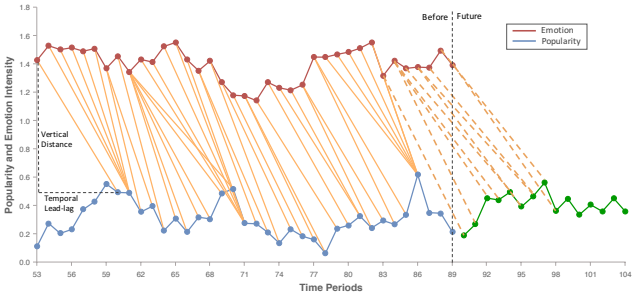

Figure 7: The prediction method of Dual ARIMA. The extended vertical distance series and temporal lead-lag series applied to predict the green points, by linking the dotted line.

6 Experiments

6.1 Performance of Sentiment Analysis Branch-Net is proposed in Section 4.1 to analyze sentiment information of short texts. To evalute the performance of the Branch-Net, we operate comparative experiments on two sentiment classification datasets. In our work, we only focus on whether the public sentiment is strong or weak, without distinguishing the positive and negative sentiments. So, the objective task of our evaluation is set as a binary classification, *strong* or *weak*.

6.1.1 Datasets

SST: Stanford Sentiment Treebank [15], an extension of movie review dataset. This dataset provides the sentiment value s varying from 0 to 1. The value of 0 presents the most negative while 1 is the most positive. In this paper, we use the binary labeling as the objective of classification. The sentiment value $0.25 \leq s \leq 0.75$ is tagged as neutral sentiment intensity and, otherwise, tagged as strong sentiment intensity.

CMRD: Cornell Movie Review Dataset [16] including 5000 subjective and 5000 objective processed sentences. The binary labeling is also set as the objective task.

6.1.2 Results and Discussion

The selected models include the extensions of Recursive neural networks like MV-RNN and RNTN and the deep neural network like CNN and FC (Fully Connected Neural Network). Another kind of models are the compound of different architecture like convolutional RNN and Branch-Net. The results of the experiments based on the SST and CMRD dataset is shown in Table 3.

From the table, we can find that the compound architecture achieves the best performance. In the

Table 3: Result of Sentiment Intensity Detection

Model	SST / %	CMRD / %
CNN [17]	86.73	92.16
FC [18]	86.24	90.79
MV-RNN [19]	82.9	-
RNTN [15]	85.4	-
conv-RNN [6]	87.71	94.13
Branch-Net	**88.13**	**94.77**

Stanford Sentiment Treebank dataset, our Branch-Net gets the largest accuracy of 88.13%. In the Cornell Movie Review Dataset, the conv-RNN model is the most outstanding model with an accuracy of 94.77%. Besides, the deep neural network like CNN and fully connected neural network performs better than the structural recursive neural networks on these two datasets.

6.2 Performance of Popularity Prediction

There have been many methods proposed to predict the popularity. However, few of these models combine the sentiment information to predict the popularity. In this experiment, we will compare the performances of these models with SENTI2POP. After that, We will also try to combine the sentiment information into non-sentiment models to see whether an improvement is achieved.

6.2.1 Dataset

In order to evaluate the performance of our model, we implement experiments on Twitter datasets. We continuously record the tweets data on Twitter for 3 months applying filtering with the keywords. In total, we collected 1.6 billion tweets on 8 topics, including *Gun Control, Trump, Immigration, AI, HIV, IOS11, Wildlife, and Air Quality.*

6.2.2 Baselines

GRU: The Gated Recurrent Unit (GRU) is proposed in [20]. GRU is quite similar to LSTM. However, the parameters of GRU are fewer than LSTM because of the absence of output gate. After the input series encoded, the hidden state h_t produced by GRU is fed into a fully connected layer, which can be viewd as a matrix transformation: $x_{t+1} = Wh_{t+1}+b$, where W is a weight matrix, b is bias. To train the GRU net, the popularity and sentiment intensity series of multiple topics is used. The GRU net is then be optimized by minimizing the MSE loss:

$$\mathcal{L} = \sum_{m=1}^{M}\sum_{t=1}^{T}(x_{t+1}^m - \hat{x}_{t+1}^m)^2 \tag{6.7}$$

CNN: The convolutional neural network (CNN) is another model widely used in time series prediction. In our task, when ignoring the sentiment information, the input is the popularity from x_{t-k+1} to x_t. The filter size is set as $1 \times m$.

When considering the sentiment information, the input is the concatenation of popularity and sentiment intensity. The filter size is $2 \times m$. The stride is set as 1 and 0-padding is applied. The output of convolution layer is fed into a pooling layer, the vertical stride is set as 2, then a fully connected layer will map the series into the prediction of x_{t+1}

Multi-layer Perceptron: In this model, a three layer perceptron machine is used to predict the time series at x_{t+1}. The input of MLP is part of the historical time series, $\{x_t, x_{t-1}, ..., x_{t-k+1}\}$. The number of perceptrons in the first layer is set to be the same as k. When considering the sentiment intensity, the first layer of the MLP is $a_h = \sum_i^I \left(\omega_i^1 P_i + \omega_i^2 S_i\right)$. Loss function is the same as Eqn. 6.7. In the last layer, softmax function is applied and the prediction of x_t+_1 is estimated.

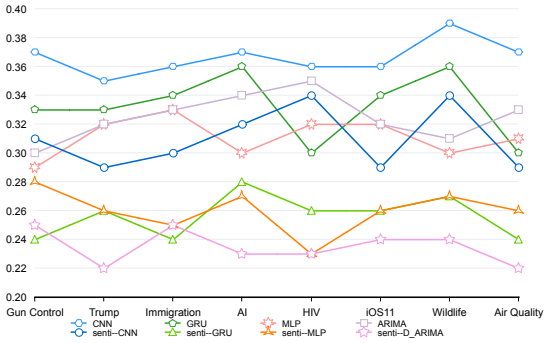

Figure 8: Results of the prediction models. The horizontal ordinate represents 8 topics. The vertical ordinate is RMSE. The Senti prefix means using sentiment information.

Visually, the prediction performance of Dual ARIMA is shown in Fig. 7. Green points are the predicted popularity. **Quantitatively**, RMSE is used as our evaluation metric. We first mark the topic popularity at the last time period as unknown. Then, we use all the methods to predict these marked popularity. The RMSE errors are calculated and visualized in Fig. 8. Sentimental Dual-ARIMA achieves the best performance in 6 out of 8 topics.

We use GRU, CNN, MLP and Dual ARIMA to implement the popularity prediction with and without the sentiment intensity, and then calculate RMSE. As is shown in Fig. 8, in the case of only utilizing popularity, the effects of prediction given by the GRU, CNN, MLP, and ARIMA models are similar.

However, after combining sentiment information,

we observe a consistent improvement of performances in all models, as is shown in Table 4.

Table 4: Percentage of RMSE improvement after combining sentiment features

CNN	GRU	MLP	ARIMA
13.98%	25.93%	10.33%	29.55%

7 Conclusions

In this paper, we designed SENTI2POP, a novel framework to predict topic popularity on social media considering sentiment intensity. Three major components in this framework were dicussed in detail: TF-TRC, a novel topic popularity metric leveraging Word2Vec and TextRank; Branch-Net, a hybrid sentiment model combining Bi-LSTM and CNN; and Dual-ARIMA, our topic popularity prediction model utilizeing wDTW-CD. Experiments confirm that sentiment intensity analysis benefits the accuracy of topic popularity prediction. Our senti2pop framework outperforms existing popularity prediction models. Possible future directions include applying this framework to other domains such as burst topic detection and stack prediciton with sentiment factor considered.

References

[1] F. Figueiredo, J. M. Almeida, M. A. Gonçalves, and F. Benevenuto, "Trendlearner: Early prediction of popularity trends of user generated content," *Information Sciences*, vol. 349, pp. 172–187, 2016.

[2] L. Hong, O. Dan, and B. D. Davison, "Predicting popular messages in twitter," in *20th international conference companion on World Wide Web*. ACM, 2011, pp. 57–58.

[3] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[4] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *2004 conference on empirical methods in natural language processing*, 2004.

[5] M. Newman, "Power laws, pareto distributions and zipf's law," *Contemp. Phys.*, vol. 46, no. 5, pp. 323–351, 2005.

[6] C. Wang, F. Jiang, and H. Yang, "A hybrid framework for text modeling with convolutional rnn," in *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 2061–2069.

[7] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *Readings in Speech Recognition*, vol. 26, no. 1, pp. 43–49, 2003.

[8] E. J. Keogh and M. J. Pazzani, "Derivative dynamic time warping," in *2001 SIAM International Conference on Data Mining*. SIAM, 2001, pp. 1–11.

[9] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.

[10] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[12] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity in phrase-level sentiment analysis," in *conference on human language technology and empirical methods in NLP*. Association for Computational Linguistics, 2005, pp. 347–354.

[13] L. Zhang, C. Aggarwal, and G.-J. Qi, "Stock price prediction via discovering multi-frequency trading patterns," in *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 2141–2149.

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[15] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[16] B. Pang and L. Lee, "A sentimental education, sentiment analysis using subjectivity summarization based on minimum cuts," in *ACL*, 2004.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[18] N. Limsopatham and N. Collier, "Modelling the combination of generic and target domain embeddings in a convolutional neural network for sentence classification." Association for Computational Linguistics, 2016.

[19] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *2012 joint conference on empirical methods in natural language processing and computational natural language learning*. Association for Computational Linguistics, 2012, pp. 1201–1211.

[20] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, 2014, pp. 1724–1734.